International Journal of Grid and Distributed Computing
Vol.7, No.2 (2014), pp.11-20
http://dx.doi.org/10.14257/ ijgdc.2014.7.2.02

A Load Balancing Algorithm based on the Variation Trend of Entropy in Homogeneous Cluster

Kehe Wu[1], Long Chen[2], Shichao Ye[2] and Yi Li[2]

[1] *Beijing Engineering Research Center of Electric Information Technology, North China Electric Power University,*
NO.2 Beinong Road, Changping District, Beijing 102206, China
[2] *Department of Control and Computer Engineering School, North China Electric Power University,*
NO.2 Beinong Road, Changping District, Beijing 102206, China
[1]*lw_ncepu@163.com,* [2] *easy_cl@163.com*

Abstract

In order to solve the problems of ill-balanced task allocation, long response time, low throughput rate and poor performance, which appeared in cluster system, this paper proposes a new load balancing algorithm of homogeneous cluster based on the variation trend of entropy. In this paper, we introduces the thermodynamic concept of entropy into load balancing algorithm and gives a new algorithm, so that we can calculate the entropy of the system to ensure that each scheduling and migration toward the entropy increasing tendency. The result of simulating experiments shows that this algorithm has more obvious advantages compared with traditional algorithms, such as achieving the load balancing status as soon as possible, shorting the task execution time, increasing system performance and so on.

Keywords: *Entropy; Variation trend; Homogeneous cluster; Load balancing*

1. Introduction

Today, with the rapid development of Internet, the increase of new applications, the exponential growth of data processing and more tasks of web services, the data flow and the calculation strength of the network, especially the core one, have increased significantly. However, single server would be unable to satisfy growing demand, so cluster system, with good scalability and high performance, has become the primary choice. As homogeneous cluster is one of the main cluster that each of the compute node in it has the same configuration of hardware and software [1], so it is significant and difficult to study how to assign task reasonably, which means that the tasks should be evenly distributed. Consequently, the load balancing mechanisms emerged, and it became a main target for resource allocation of cluster system.

In recent years, the load balancing technology is developing very fast in automation, intelligence and exactness prospect. More and more known load balancing information and unknown predictive information have been chosen by researchers as the judgment standard of load balancing. Mohammed A M Ibrahim, *et al.*, [2] proposes the dynamic load balancing algorithm based on round robin scheduling, so that all nodes in the cluster can be equally selected in a reasonable order, which can solve the problem of the parallel search tree well. However, since this algorithm do not take the current moment load of the nodes into consideration, the judgment of the dynamic load balancing is not precise enough. Dai Y, et al

ISSN: 2005-4262 IJGDC

[3] applied the idea of fuzzy control and heuristic strategy from the traditional control theory to the load balancing, and proposes a fuzzy control-based heuristic algorithm for load balancing in workflow engine, which offers a new idea for solving the problem of load balance. Liu Z H, *et al.*, [4] gives a load balancing optimization algorithm based on genetic algorithm. However, this algorithm can only be used in laboratory but unable to meet the complex needs of practical applications. Balasubramaniam M, *et al.*, [5] presents a dynamic load balancing library in heterogeneous cluster, which can't well satisfy the demand of homogeneous cluster. The relationship between the load balancing problems of cluster system and the energy is discussed by Dong J Y, *et al.*, [6], from aspects of entropy and generalized complexity, they calculate the value of the energy, which could make the cluster system to achieve an equilibrium state, and it makes possibility for solving the problem of load balancing by using the change of entropy.

Therefore, from what present above, this paper proposes a load balancing algorithm based on the variation trend of entropy in homogeneous cluster, and it is used to equally distribute the tasks in homogeneous cluster to reduce the response time and increase the throughput, so that it would improve the performance of cluster system. The paper is organized as follows: In Section 2, we describe the basic concept of load balancing and entropy; Section 3 we introduce the thermodynamic concept of entropy into load balancing algorithm, define the model and then analyze the basic properties of it; In Section 4, we present our algorithm and give a brief introduction to it. Section 5 is about some experiments on this algorithm and others. The last section is the conclusion of the paper and gives lights to our future work.

2. Basic Concept

Before introducing the algorithm, it is necessary to learn about the basic theory of entropy and load balancing, so as to understand the algorithm better.

2.1. Concept of Entropy

The concept of entropy originated in physics, by the German physicist Rudolf Clausius first proposed in 1850 [7]. It is primarily used for representing the evenness degree of any kind of energy distribution in the space, the more uniform energy distribution, the greater the entropy. When the energy of a system achieves a completely uniform distribution, the entropy of the system reaches a maximum. We can describe entropy as follows:

$$\Delta S = \frac{Q}{T} \tag{1}$$

where Q denotes the change of system energy and represents the variation of entropy.

2.2. Concept of Load Balancing

The load is an abstract concept to indicate how busy the system is and it refers to the subtasks, which assigned to each server node and executed in parallel. The so-called load balancing strategy means to balance the load of each server node by adopting certain policy to make the load essentially equal. It can be understood in two ways: On the one hand, it refers to allocate the large volumes of concurrent access and data traffic to multiple server nodes, and process them separately to reduce the waiting time; On the other hand, it means that a single heavy load calculating task can be shared to multiple server nodes for dealing with in parallel, and then summarizes the results back to the user, improving the treatment capacity of the system [9]. The mathematical model is defined as follows:

Definition 1 The so-called load-balancing means giving a set of load $L = \{L_1, \dots, L_n\}$, a set of server nodes $S = \{S_1, \dots, S_m\}$ and a set of current server load $SL = \{SL_1, \dots, SL_m\}$, to find a function $f(L)$, which the set of load L can be mapped to the set of server nodes S, making the load SL_i of each server node S_i be essentially equal, that is:

$$SL_1 \cong SL_2 \cong \cdots \cong SL_m \tag{2}$$

Among this, SL_i represents the sum of all the load $f(L_i)$ mapped to this server node S_i.

If use τ_o to reflect the time needed for executing task L_o on the server node S_i, so the time needed for executing all the task on the server node S_i is as follows:

$$t_i = \sum_{o \in f(L_i)(i=1,\dots,n)} \tau_o \tag{3}$$

Definition 2 If m equals 1, that means there is only one server node, and all the tasks should be executed serially on the server node, so the time needed is the sum of all the time, which can be represented as T_1 shown below:

$$T_1 = \sum \tau_o \ (o = 1, \dots, n) \tag{4}$$

Definition 3 If m is greater than 1, that means there is more than one server node, and the tasks can be shared to multiple server nodes for dealing with in parallel, the time needed is represented as T_m shown below:

$$T_m = \max_{i=1,\dots,n} t_i \tag{5}$$

Thus, the goal of the load balancing is to solve the mapping $f(L)$ to get the minimum of T_m in case of $SL_1 \cong SL_2 \cong \cdots \cong SL_m$.

3. Model of the Algorithm

Through the above description of the basic concept, we find that the features of load-balancing in cluster system have much in common with the relative concepts of thermodynamics system [10], which means that the entropy can be used to indicate the randomness of material, the more homogeneous the bigger the entropy, while the purpose of load-balancing is the load-distribution uniformity, so we introduce the concept of entropy into the cluster system, and then redefine some concepts of load balancing as follows:

Definition 4 (the concept of entropy) If a homogeneous cluster system has n compute nodes, and at time t, the load of the node K is L_k as well as the relative load factor is $q_k = L_k / \sum_i L_i (i = 1, \dots, n)$, then the system entropy value $H(t)$ at time t can be expressed as below:

$$H(t) = \sum_{k=1}^{n} [q_k * \ln(1/q_k)] \tag{6}$$

Definition 5 (the goal of load-balancing) The goal of load balancing is always towards the trend of increasing entropy, the greater the entropy, the more homogeneous the load, and when the load of cluster system completely uniform distribution, the entropy value reaches the maximum at the same time. That is, the changing trends of the entropy value determine the load-balancing.

By the above definition, we can conclude that the entropy of homogeneous cluster has the following properties:

Property 1 The load balancing is always towards the trend of increasing entropy, that is, the changing trends of the entropy value determine the load-balancing.

As the entropy can be used to indicate the randomness of material, the increasing of the entropy represents material tends to be stable as well as the goal of load-balancing is the average load distribution, so the increase of the entropy can achieve the goal, which means that the changing trends of the entropy value can determine the load-balancing.

Property 2 The entropy will reach its maximum if and only if the load is completely balanced, that is, the state of maximum entropy is the state of load most balanced.

By property 1 we know that the load balancing is always toward the trend of increasing entropy, so when the load completely balanced, the entropy reached its maximum.

Property 3 The entropy reaches its maximum, the execution time of the task reaches the minimum.

Property 4 The change of the entropy takes on the rule that increases at first until reaching a maximum after a period of time, and then decreases late.

Through the above definition and properties, we can realize that entropy is a good measure for judging the degree of load balancing. In this paper, a load balancing algorithm based on the variation trend of entropy is proposed, which can make the system load balanced in a short period of time towards the trend of entropy increase, making full use of server resources and avoiding the waste caused by the uneven distribution of the load.

4. Implementation of the Algorithm

In order to achieve the goal of load balance in cluster system, the operation can be divided into two stages: the first one is load scheduling, which spreads the load equally across all servers in the cluster, so that it can make the system to achieve balance equilibrium; the second one is migration, making partial adjustment after the load distribution on the servers, which means migrating the tasks on the overload server nodes to a lightly one. Therefore, the following four questions should be solved:

1) The collection and processing of load information on server nodes
2) The selection of the scheduling policy
3) The choice of the migration strategy
4) The implementation of migration

In this paper, a new load balancing algorithm based on the variation trend of entropy in homogeneous cluster was put forward and solved the above difficult problems. The solution of the four problems are described in detail below, and finally, a complete description of the algorithm is given.

4.1. Collection and Processing of Load Information

This paper focus on the variance of entropy. According to the definition of entropy, it is related to the relative load factor, which is the ratio of the load of the nodes to the total load of the system. And the load can not only be calculated by the number of tasks, but also can be measured by the calculation of tasks. However, in the homogenous cluster system, when the calculation of tasks can be measured, it is better to use the total computation of tasks to reflect

the load of the server node Si than the number of tasks, so we chose the calculation of tasks as a measure of the load for the server node in this paper. In order to calculate the value of entropy, we need to know the load of each server node, which means that we need to be synchronized, coordinate the nodes status information to the back-end services, and it could cause a lot of traffic and increase the load of the system by using traditional dynamic monitoring, so in this paper, introduced a monitor node to achieve centralized collection of the load information, and then calculated the relative load factor q_k of each server node as well as the entropy $H(t)$ of the current system. Finally, the all information we obtained should be feedback to the scheduler for load scheduling. In this way, each server node merely need to transmit the load information to the monitor node for uniformly handing and feedback the information, which ensured the state synchronization between servers and reduced the traffic. Meanwhile, to avoid single point of failure caused by a single monitor node with a hot-standby strategy, which means that it can switch over to the standby monitor node when the main node failure to ensure the system running normally.

4.2. Selection of the Scheduling Policy

There are many mature scheduling algorithms, such as, Round-Robin Scheduling Algorithm, Least-Connection Scheduling Algorithm, Weighted Round-Robin Scheduling Algorithm, Destination Hashing Scheduling Algorithm and so on. The algorithm proposed in this paper is based on the variation trend of entropy, which means scheduling based on the change of the entropy. After the scheduler obtained relative load factor of each server node and the entropy of system from the monitor node, we need to calculate the changes of the entropy in accordance with the calculation of the scheduled tasks assuming that the tasks have been scheduled to the server node before we do the schedule, and then, selected a server node, which the entropy of it increased with the maximum increment, as the target of the task scheduling machine to complete this schedule. In this way, the entropy of the system is increased with the maximum increment after every scheduling, so that the entropy increased to the maximum when we finished the task scheduling, and according to above Property 2, the load of the system have achieved the equilibrium state at the moment. This scheduling algorithm can make the system to achieve load balancing state in a comparatively short period of time so as to avoid wasting server resources.

4.3. Choice of the Migration Strategy

Due to the differences of the task and the time needed to complete the task is different, which means some tasks completed, while some tasks are still performed, it leads to the change of the relative load factor of the server node, which will influence the changes of the entropy. According to Property 4, we know that the entropy will be reduced and the load of the system become unbalanced at this time, and we need to do something to make it back into balance, such as, the migration. However, the migration need to take up system resources and time, and if we do it without rules will do more harm than good, which will not only improve the system load, but will increase the burden on the system.

Therefore, the key point we have to focus on is when and how to do the migration. In this paper, we propose a load balancing algorithm based on the variation trend of entropy, so we take the entropy as the judgment condition of whether to do the load migration. For this, we design a threshold value H° of system entropy as the load migration critical conditions and compare the current system entropy $H(t)$ calculated by the monitor node with this threshold value H°. If $H(t)$ is less than H°, it reflects that the load of system is unbalanced, we need to

do the load migration, which means transferring the tasks from high-load nodes to low-load nodes, to balance the load of each server node as well as to increase the entropy value, so as to realize the system load balancing; On the country, if $H(t)$ is greater than or equal to H°, it explains that the load of system is balanced, there is no need to do the migration. It can be seen that the key influence factors for migration is the threshold value H° of system entropy. If the threshold value H° is too low, it will reduce the chances of migration, and that will lead to the load unbalanced, meaning the tasks on the high-load nodes can't be processed, while there is no task for the low-load nodes. As a result, there will be a waste of system resources, and even a system crash caused by the too heavy load; otherwise, if the threshold value H° is too high, the migration will frequently happened, and the migration need to consume the system resources, so that it will degrade system performance.

4.4. Implementation of the Migration

The monitor node calculates the relative load factor by obtaining the current load information of each server node, so as to calculate the current entropy value $H(t)$ of the system, which will be compared with the threshold value H°, and then the results may feed back into the scheduler, finally, the scheduler determines whether to do the migration. The migration is also based on the entropy value, to ensure that the entropy of the system would increase after the migration, otherwise, the migration is not executed. This is an NPC [11] problem, which can't find a common formula to solve it, so we do the following conventions:

(1) The tasks are always migrated from high-load nodes to low-load nodes, but not vice versa. It means that the migration is always toward the increasing trend of system entropy value.

(2) The tasks with too small calculating amount are beyond the select scope of the migration because the migration need to take up system resources and time, and the time taken to migration is far beyond the execution time of tasks. If we do the migration, there will be a waste of system resources and time.

Therefore, whether the system do the migration depends on whether the system entropy value increased and the load-balanced situation improved significantly after the migration.

4.5. Description of the Algorithm

The main description of this algorithm is listed as follows:

(1) According to the collected information, we can calculate the load L_k and relative load factor q_k of each server node as well as the average relative load factor $q^{\circ} = 1/n$, so as to obtain the system's current entropy value $H(t)$. Then, turn to step (2);

(2) To compare the system's current entropy value $H(t)$ with the threshold value H°:

① If the $H(t)$ is less then H°, it means that the system load is unbalanced and need of migration. For the server nodes that their relative load factor q_k of each server node is greater than or equal to the average relative load factor q°, should be sorted by the relative load factor q_k in descending order to form a server queue Q_s, that is the server which needed to do the load migration; for the other server nodes that their relative load factor q_k of each server node

is less than the average relative load factor q°, should be sorted by the relative load factor q_k in ascending order to form a server queue Q_t, that is the server where the load migrated to, turn to step (3) to do the migration;

② If the $H(t)$ is greater than or equal to H°, it shows that the system load is balanced and it is unnecessary to do the migration. To see if there is any new task to be scheduled and if there be, turn to step (4); otherwise, return to step (1);

(3) To traverse the queue Q_s and Q_t, taking out the nodes S and T to migrate the task on node S to node T, ensuring that the entropy of the system would increase after the migration. For the node S, the operation will be finished until the relative losad factor q_k of each server node is less than the average relative load factor q°, then remove the node S from the queue Q_s; For the node T, the operation will be finished until the relative load factor q_k of each server node is greater than or equal to the average relative load factor q°, then remove the node T from the queue Q_t, and the migration will be finished until any one of the queue is empty, return to step (1);

(4) To schedule the new tasks, forming a server queue Q sorted by the relative load factor q_k in ascending order, scanning this queue and finding out the node S, which its entropy increased with the maximum increment after the scheduling. If it exists, then scheduled the task to the node S to finish the task scheduling, return to step (1).

5. Experiments and Results

The test environment is a homogeneous cluster comprised by five servers with the same configuration of 2.4GHz CPU, 8G RAM and Ubuntu 12.04. We did the load scheduling and load balancing on the servers, which were indicated by the letter A, B, C, D and E. Among a large number of experiments, this paper selects the most representative one to explain.

In this experiment, the experiment objects are fifty tasks selected randomly from system, and the environment is the homogeneous cluster formed of five servers. This experiment is divided into two steps: firstly, adopting the traditional algorithm to schedule and load balance, and record the load amount of system's each servers after reaching balanced, and then calculate the corresponding entropy; secondly, do the same task by using the algorithm mentioned in this paper; finally, compare the results of two groups and make an analysis of it.Table1 shows the load of each server after scheduling with the two algorithms, the new algorithm and the traditional one. For the convenience this paper has the calculation of the task instead of the load.

Table 1. The Load of each Server after Scheduling with the Two Algorithms

Nodes	the load after scheduling	
	Use the traditional algorithm	Use the new algorithm
Node A	11	10
Node B	11	13
Node C	16	12
Node D	15	15
Node E	10	13
The system entropy value	1.5914	1.6010

By applying to the definition of entropy, which is $H(t) = \sum_{k=1}^{n}[q_k * \ln(1/q_k)]$, the current entropy of system can be calculated, and the entropy with the traditional algorithm is 1.5914, while the entropy with the new algorithm is 1.6010. According to the properties of the entropy, we know that the scheduling with the new algorithm can make the system load more balanced than the one with the traditional algorithm, that is, the new algorithm can make the system load tend to be balanceable faster than the traditional one.

Table 2 shows the load of each server after migration with the two algorithms, the new algorithm and the traditional one.

Table 2. The Load of each Server after Migration with the Two Algorithms

Nodes	The load after migration	
	Use the traditional algorithm	Use the new algorithm
Node A	13	12
Node B	11	13
Node C	13	12
Node D	13	13
Node E	13	13
The system entropy value	1.6074	1.6087

The current entropy of system can be calculated, and the entropy with the traditional algorithm is 1.6074, while the entropy with the new algorithm is 1.6087. By comparing with the above data, both algorithms can make the system entropy to increase after migration, which means that the system load distributed more evenly after migration. Moreover, the entropy value with the new algorithm is greater than the one with the traditional algorithm, showing that the new algorithm can make the system load balancing more effective than the traditional one.

6. Conclusion

By introducing the concept of entropy in thermodynamics into load balancing, this paper proposes a load balancing algorithm based on the variation trend of entropy in homogeneous cluster, gives the theoretical model and the basic properties of system entropy, uses the entropy as a measure for the degree of load balancing, and schedules and migrates in accordance with the entropy change to make the system achieving the goal of load balancing faster and better. Meanwhile, this paper compares the algorithm with other traditional algorithm, showing that the new algorithm is better than the traditional algorithm on the time and degree of system load balancing, which indicates that the new algorithm is workable to a certain extent to balance the load in homogeneous cluster.

However, since the algorithm is conducted on the homogeneous cluster without consideration of the heterogeneous of servers, there still have many problems needed to be solved in practical application environment. Therefore, the next job of this paper should focus on the optimization of this algorithm that enable it to adapt the situation of heterogeneous cluster, as a result , to obtain a load balancing algorithm with wide applicability.

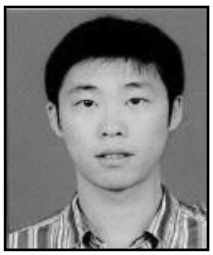
References

[1] K. Y. Lu and H. Y. Sun, “Parallel computing entropy in homogeneous cluster”, Journal of Shenzhen University Science and Engineering, vol. 1, no. 26, **(2009)**.

[2] A. M. I. Mohammed and X. D. Lu, "Performance of dynamic load balancing algorithm on cluster of workstations and PCs", Proceedings of 5th International Conference on Algorithms and Architectures for Parallel Processing, Beijing, China, **(2002)** October 23-25.
[3] Y. Dai and J. Cao, "Fuzzy control-based heuristic algorithm for load balancing in workflow engine", Journal on Communications, vol. 11, no. 27, **(2006)**.
[4] Z. H. Liu and X. L. Wang, "Load balancing algorithm with genetic algorithm in virtual machines of cloud computing", Journal of Fuzhou University(Natural Science Edition), vol. 4, no. 40, **(2012)**.
[5] S. C. Chauu and W. C. F. Ada, "Load balancing between computing clusters", Proceedings of the 4th International Conference on Parallel and Distributed Computing, Beijing, China, **(2003)** August 27-29.
[6] J. Y. Dong, P. Wang and Y. Chen, "The Entropy Value of Load Blance Algorithm about Cloud Computer Cluster", Journal of Chengdu University of Information Technology, vol. 6, no. 25, **(2010)**.
[7] Entropy. EB/OL. http://en.wikipedia.org/w/index.php? title=Entropy&oldid=572347756, **(2013)**.
[8] H. Zhou and S. X. Luo, "The Computer Cluster and the Parallel Computer Environment based on the Networks", Computing Techniques for Geophysical and Geochemical Exploration, vol. 4, no. 23, **(2001)**.
[9] Y. J. Luo, X. L. Li and R. X. Sun, "Summarization of the Load- balancing Algorithm", Scitech Information Development & Economy, vol. 23, no. 18, **(2008)**.
[10] H. Y. Sun, W. X. Xie and X. Yang, "A Load balancing algorithm based on Parallel computing entropy in HPC", Journal of Shenzhen University Science and Engineering, vol. 1, no. 24, **(2007)**.
[11] A. Heirich, "Analysis of Scalable Algorithms for Dynamic Load Balancing and Mapping with Application to Photo-realistic Rendering", Caltech, Pasadena, **(1998)** October 11-14.

Authors

Kehe Wu, He received his M.Sc. in Mechatronics (1995) and PhD in Computer Sciences (2005) from North China Electric Power University. He is currently a professor in information security in North China Electric Power University. He is a Director of the Chinese Association for artificial intelligence and a Senior Member of China Electric Power Information Standardization Committee. His current research interests include different aspects of Intelligent Software, Cloud Computing and Information Security.



Long Chen, He received his BS degree in computer science and technology from North China Electric Power University, Beijing, China, in 2010. He is now studying the PhD of computer application technology in North China Electric Power University, Beijing, China. His current research interests include Cloud Computing, Resource Scheduling and Information Security.

Shichao Ye, He received his BS degree in Computer software and theory from North China Electric Power University, Beijing, China, in 2009. He is now studying the PhD of information security in North China Electric Power University, Beijing, China. His current research interests include Computer Theory and Information Security.

Yi Li, He received his BS degree in computer science and technology from North China Electric Power University, Beijing, China, in 2010. He is now studying the PhD of computer application technology in North China Electric Power University, Beijing, China. His current research interests include Cloud Computing, Distributed Storage and Networking.

